

A geometric approach for classification and comparison of structural variants

Suzanne Sindi^{1,2}, Elena Helman³, Ali Bashir⁴ and Benjamin J. Raphael^{2,3,*}

¹Division of Applied Mathematics, ²Center for Computational Molecular Biology, Brown University, Providence, RI,

³Department of Computer Science and ⁴Bioinformatics Graduate Program, University of California, San Diego, CA, USA

ABSTRACT

Motivation: Structural variants, including duplications, insertions, deletions and inversions of large blocks of DNA sequence, are an important contributor to human genome variation. Measuring structural variants in a genome sequence is typically more challenging than measuring single nucleotide changes. Current approaches for structural variant identification, including paired-end DNA sequencing/mapping and array comparative genomic hybridization (aCGH), do not identify the boundaries of variants precisely. Consequently, most reported human structural variants are poorly defined and not readily compared across different studies and measurement techniques.

Results: We introduce Geometric Analysis of Structural Variants (GASV), a geometric approach for identification, classification and comparison of structural variants. This approach represents the uncertainty in measurement of a structural variant as a polygon in the plane, and identifies measurements supporting the same variant by computing intersections of polygons. We derive a computational geometry algorithm to efficiently identify all such intersections. We apply GASV to sequencing data from nine individual human genomes and several cancer genomes. We obtain better localization of the boundaries of structural variants, distinguish genetic from putative somatic structural variants in cancer genomes, and integrate aCGH and paired-end sequencing measurements of structural variants. This work presents the first general framework for comparing structural variants across multiple samples and measurement techniques, and will be useful for studies of both genetic structural variants and somatic rearrangements in cancer.

Availability: <http://cs.brown.edu/people/braphael/software.html>

Contact: braphael@brown.edu

1 INTRODUCTION

Characterizing the DNA sequence differences that distinguish individuals is a major challenge in human genetics. Until recently, these differences were thought to be mostly single nucleotide changes. There is increasing appreciation for the prevalence of structural variation, including duplications, deletions and inversions of large blocks of DNA sequence, in the human genome (Sharp *et al.*, 2006). Structural variants have recently been linked to diseases such as autism (Marshall *et al.*, 2008), and cataloging these variants is an important step in determining the genetic basis of disease.

Structural variants typically span thousands of nucleotides and are more difficult to define than single nucleotide polymorphisms (SNPs). Two techniques have been used to identify structural variants in the human genome: *array comparative genomic hybridization (aCGH)* and *end sequence profiling (ESP)*, also called *paired-end mapping*. Both of these techniques were first developed for the analysis of somatic structural rearrangements in cancer genomes (Pinkel and Albertson, 2005; Pinkel *et al.*, 1998; Volik *et al.*, 2003) and later applied to discover structural variation in normal genomes (Iafate *et al.*, 2004; Kidd *et al.*, 2008; Korbel *et al.*, 2007; Sebat *et al.*, 2004; Tuzun *et al.*, 2005). In aCGH, differentially fluorescently labeled DNA from test and reference genomes are hybridized to an array of genomic probes derived from the reference genome. Measurements of test:reference fluorescence ratio at each probe identifies locations of the test genome that are present in higher or lower copy in the reference genome. This technique detects copy number variants but is blind to copy neutral variants such as inversions. In paired-end mapping approaches, DNA fragments, or clones, from a test genome are sequenced from both ends, and these sequences are mapped to a reference genome sequence. Pairs of end sequences, called *end sequence pairs* or *mate pairs*, with discordant mappings identify inversions, translocations, transpositions, duplications and deletions that distinguish the test genome from the reference genome. Next-generation DNA sequencing technologies such as those from Illumina, Applied Biosystems and 454 Life Sciences now make it possible to apply this approach to a large number of individuals. We distinguish paired-end mapping from whole-genome assembly, which would provide the ultimate dataset for studies of structural variation (Levy *et al.*, 2007; Wheeler *et al.*, 2008), but remains cost prohibitive for a large number of human genomes.

Both aCGH and paired-end mapping do not precisely identify the boundaries, or *breakpoints*, of the measured variant. In aCGH, a breakpoint is localized only to the distance between the genomic probes straddling the copy number change, while in paired-end mapping the localization depends on the number and size of fragments that span the variant (Fig. 1A). There are no standard methods for identifying the boundaries of structural variants in paired-end mapping studies and different heuristics have been used (Kidd *et al.*, 2008; Korbel *et al.*, 2007). Remarkably, despite ambiguity resulting from both measurement and analysis, published studies of structural variants report the breakpoints to single nucleotide resolution without revealing the measurement uncertainty, or ‘error bars’, in the localization of the variant. Consequently, the existing databases of human structural variants

*To whom correspondence should be addressed.

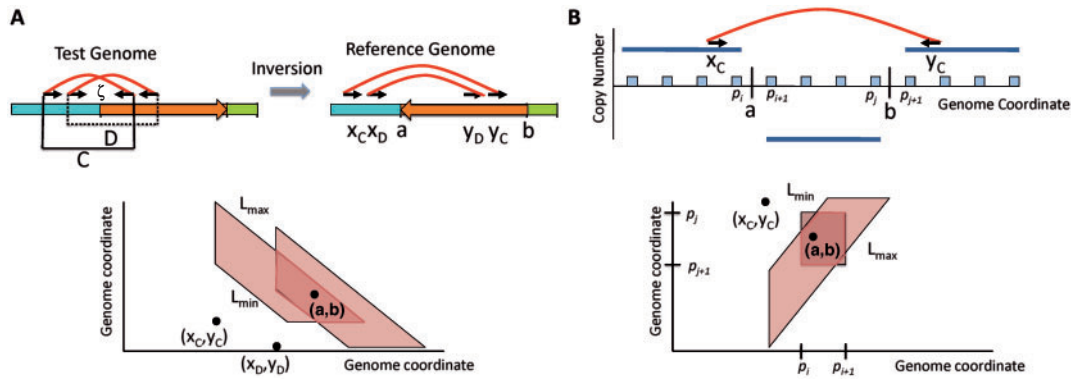


Fig. 1. Derivation of regions of uncertainty (*breakpoint regions*) from ESP and aCGH data. **(A)** (Top panel) In ESP, or paired-end mapping, both ends of a fragment of a test genome are sequenced and aligned to the reference genome. Here, alignment of ends of fragments *C* and *D* yields ES pairs (x_C, y_C) and (x_D, y_D) on the reference genome that suggest an inversion. (Bottom panel) The intersection of breakpoint regions defined by Equation (1) indicates the possible locations of inversion breakpoints *a* and *b* that are consistent with the ES pairs. **(B)** (Top panel) In an aCGH experiment, the reference genome is segmented into regions of equal copy number according to measurements at genomic probes (boxes). A deletion with breakpoints *a* and *b* is identified as a change in copy number between probes p_i and p_{i+1} and between probes p_j and p_{j+1} . (Bottom panel) The intervals $[p_i, p_{i+1}]$ and $[p_j, p_{j+1}]$ define a rectangular breakpoint region. This region is intersected with the breakpoint region defined by an ES pair (x_C, y_C) to refine the locations *a* and *b* of the deletion.

(Iafrate *et al.*, 2004) also do not contain information on the uncertainty of the breakpoints.

Each new study of structural variation compares the newly discovered variants to those previously reported. In addition to the ambiguities described above, there is also the problem of deciding when two variants (perhaps measured via different approaches) are the same. The usual approach is to define two variants to be the same if they are ‘near’ each other, where near is defined using an arbitrary and study-dependent threshold. With such an approach, there is no assurance that the two variants are indeed the same, or are merely closely located on the genome. The situation is further exacerbated by reports that different human structural variants may overlap or have multiple states (Perry *et al.*, 2008; Scherer *et al.*, 2007), and that recurrent (but not identical) variants may exist at the same locus.

Standard methods for defining structural variants and publicly available tools for comparing structural variants across different studies are urgently needed. Two recent works introduced more refined approaches for analysis of structural variants and are promising steps in this direction. Lee *et al.* (2008) describe a probabilistic method for resolving ambiguities in mapping end-sequenced fragments using the distribution of fragment lengths in a single sample. Bashir *et al.* (2008) estimate the probability that paired-end sequenced clones from cancer genomes contain fusion genes and explicitly incorporate the uncertainty in measurement of rearrangement breakpoint into their calculation. Neither of these approaches address the comparison of variants across multiple samples, and are further limited in their handling of measurement uncertainty and consideration of all classes of structural variants, respectively.

Here, we introduce a general geometric framework for classification and comparison of structural variants. Our approach provides a principled way to cluster multiple measurements of a variant in a single sample and to compare variants across samples. We explicitly model the underlying measurement uncertainty of both paired-end mapping (from both older and next-generation sequencing technologies) and aCGH. We represent the uncertainty in the measurement of a structural variant, which we refer to as

the *breakpoint region*, as a polygon in the plane. We formulate the problems of comparing variants as computing all intersections and maximal intersections of breakpoint regions. These formulations allow the user to examine conserved variants at varying levels of granularity, instead of only producing a single best cluster of overlapping variants. We derive an efficient plane sweep algorithm from computational geometry to compute these intersections.

We demonstrate our Geometric Analysis of Structural Variants (GASV) program with three applications. First, we apply our method to recent paired-end sequencing studies of nine human individuals. We show that GASV identifies rearrangement breakpoints with high precision. In dozens of cases, we localize rearrangement breakpoints to < 2.5 kb by combining the measurements from ≈ 40 kb clones across multiple individuals. In the most extreme example, eight end-sequenced clones from four different individuals localize the inversion breakpoints to within 286 bp. Such precise localization was not reported in the original published analysis of these nine individuals. Moreover, we show that the published locations of many variants are different from the breakpoints supported by the data. Second, we perform a comparative analysis of variants from the nine normal individuals with variants identified in paired-end sequencing of several cancer samples. We find that a significant fraction (5–53%) of rearrangements identified in the cancer genomes are consistent with inversion and deletion variants found in the normal genomes. Finally, we show how GASV integrates both aCGH and paired-end sequencing measurements of variants in three cancer genomes.

Our geometric method for multi-sample and multi-platform identification and comparison of structural variants should prove useful for studies of human structural variation such as the 1000 Genomes Project and for cancer genome sequencing studies such as The Cancer Genome Atlas.

2 METHODS

Consider a *reference genome* represented as a single interval *G* (i.e. we concatenate multiple chromosomes) and a closely related *test genome*. We define a structural variant to be a difference between a *test genome*

and *reference genome* that is due to a rearrangement resulting from DNA breakage followed by a aberrant repair or insertion of a new DNA. Structural variants include inversions, translocations, transpositions, and insertions/deletions. Each of these variants is thus associated with a set of breakpoints where DNA breaks and/or repair occurs. For example, an inversion is a result of the reference genome being cut at two genomic coordinates, a and b , and the DNA segment between a and b flipped in the test genome so that the nucleotide at position $a-1$ is adjacent to the nucleotide at position b and a is adjacent to $b+1$ (Fig. 1A). Similarly, a deletion is defined by coordinates a and b in the reference such that $a-1$ is joined to $b+1$ in the test genome (Fig. 1B). Note that this is a simplification of the underlying biology, as there are sometimes small insertions or deletions at breakpoints, but these small changes have limited effect on the analysis of larger structural variants.

2.1 Breakpoint regions and variant uncertainty

Neither paired-end mapping nor aCGH measure the breakpoints of a structural variant exactly. Rather, each technique localizes breakpoints to a region of the reference genome, which we refer to as the *breakpoint region*. We describe the derivation of this region for each of these experimental techniques.

2.1.1 Paired-end mapping In the paired-end mapping, or ESP, fragments of genomic DNA from a test genome are sequenced from both ends, and the resulting pair of end sequences are aligned to the reference genome. We assume that each fragment¹ C has ends that map uniquely to the reference genome. Thus, each fragment C corresponds to a pair of locations in the reference genome where the end sequences map. An end sequence may align to either DNA strand, and so each mapped end has a sign (+ or -) indicating the mapped strand. We call such a signed pair (x_C, y_C) an end sequence pair (*ES pair*), where by convention $|x_C| < |y_C|$. Typically, the length of the fragment, L_C , is known to lie within a range $[L_{\min}, L_{\max}]$. Fragment sizes range from ≈ 150 kb for BAC clones to a few hundred base pairs for next-generation sequencing methods. We say that a ES pair is a *valid pair* (Raphael et al., 2003) if the ends have opposite, convergent orientations and the distance between the mapped ends is within the range of fragment lengths: i.e. $(+x_C, -y_C)$ is valid if $L_{\min} \leq |y| - |x| \leq L_{\max}$. Otherwise, if the ends have abnormal distance or orientation, we say that the pair is an *invalid pair*. Invalid pairs indicate putative genome rearrangements or possibly mapping/assembly errors.

For concreteness, consider the case of a test genome that differs from the reference genome by a single inversion with breakpoints a and b (Fig. 1A) that fuse at a single coordinate ζ in the test genome. A fragment C from the test genome with length between L_{\min} and L_{\max} and containing ζ is end-sequenced. The resulting ES pair (x_C, y_C) will be an invalid pair indicating that C is not a contiguous piece of the reference genome (Fig. 1A). The invalid pair (x_C, y_C) does not uniquely identify the breakpoint (a, b) . However, if we assume that: (i) only a *single* breakpoint is contained in the fragment C ; and (ii) $a > x_C$ and $b > y_C$ (without loss of generality); then the length L_C of C is equal to $(a - x_C) + (b - y_C)$. Thus, a breakpoint (a, b) that is consistent with (x_C, y_C) must satisfy

$$L_{\min} \leq (a - x_C) + (b - y_C) \leq L_{\max}. \quad (1)$$

We define the *breakpoint region* $B(C)$ of an invalid fragment C to be the breakpoints (a, b) satisfying the above equation. The constraint (1) has a straightforward geometric interpretation: if we plot an invalid pair (x_C, y_C) as a point in the 2D space $G \times G$ then the breakpoint region defines a trapezoid (Fig. 1A). We emphasize that a and b cannot be chosen independently; doing so corresponds to defining the breakpoint region to be a rectangle, and allows breakpoints that give insert sizes outside the allowed range $[L_{\min}, L_{\max}]$.

¹We use the term *fragment* to describe both genomic clones (BACs, fosmids, plasmids) used by older sequencing technologies and DNA fragments in the mate-pair libraries employed in next-generation sequencing technologies.

If another fragment D contains the same fusion point ζ , then the corresponding breakpoint (a, b) lies within the intersection $B(C) \cap B(D)$ of the trapezoids $B(C)$ and $B(D)$ (Fig. 1A). Conversely, we will assume that if the trapezoids defined by several invalid pairs intersect, then they share a common breakpoint. As the number of fragments that are end-sequenced increases, more fragments will contain the same fusion point and the area of the intersection of breakpoint regions will decrease. Thus, the uncertainty in the location of the breakpoint (a, b) decreases. We define a *cluster* to be a set of fragments whose breakpoint regions have non-empty intersection.

The description above generalizes to other types of structural variants including translocations, insertions, deletions and transpositions. For example, invalid pairs with $\text{sign}(x_C) = \text{sign}(y_C) = -$ also indicate inversions (corresponding to the other fusion point), while invalid pairs with $\text{sign}(x_C) = +$ and $\text{sign}(y_C) = -$ indicate insertions or deletions. Fragments with ends mapped to different chromosomes indicate translocations. As above, we assume that the fragment C contains only a *single* breakpoint. The breakpoints (a, b) that are consistent with the invalid pair (x_C, y_C) satisfy the inequalities

$$L_{\min} \leq (\text{sign}(x_C)a - x_C) + (\text{sign}(y_C)b - y_C) \leq L_{\max} \quad (2)$$

This equation generalizes (1) and is summarized by the rule: ‘end sequences point toward the breakpoint’.

2.1.2 Array comparative genomic hybridization In aCGH, a *breakpoint region* is defined as the genomic interval between the two adjacent probes p_i and p_{i+1} that define the endpoints of segments with unequal copy number (Fig. 1B). A pair of such breakpoint regions (e.g. those resulting from a deletion) give two intervals $U = [p_i, p_{i+1}]$ and $V = [p_j, p_{j+1}]$ that define a rectangle $U \times V$ in 2D space $G \times G$. This rectangle determines the locations of breakpoints $(a, b) \in U \times V$ consistent with the segmentation. Note that in addition to fragments that span deletions (Fig. 1B), the boundaries of aCGH segments often indicate the locations of other types of rearrangements including translocations (Aerni et al., 2009; Campbell et al., 2008).

2.2 Efficient computation of overlapping breakpoint regions

Given a set B_1, \dots, B_n of breakpoint regions, our goal is to identify subsets of intersecting breakpoint regions. Such a subset suggests these breakpoint regions are multiple measurements of the same structural variant. In addition, we want to identify all such regions of intersection, and to label these by the breakpoint regions that are part of the intersection. We formalize these problems as follows:

ALL INTERSECTIONS OF BREAKPOINT REGIONS. *Given a set $\mathcal{B} = \{B_1, \dots, B_n\}$ of breakpoint regions, identify and label all non-empty intersections of subsets of \mathcal{B} .*

Since each breakpoint region B_i is a convex polygon (trapezoid or rectangle), the solution to the above problem relies on computing intersections of convex polygons, a well-known problem in computational geometry (Preparata and Shamos, 1985). A naive brute-force approach that checks all 2^n subsets of \mathcal{B} for intersection is very inefficient. Moreover, a single breakpoint region can have distinct intersections with different subsets of other breakpoint regions (Fig. 2). Thus, it is not sufficient to consider only pairwise intersections or iteratively merge breakpoint regions to existing intersections. Below, we describe an efficient plane sweep algorithm that solves the ‘All Intersections Problem’.

While the ‘All Intersections Problem’ provides the most comprehensive description of the overlaps between breakpoint regions, the output can be quite large since the number of regions of intersections grows rapidly as n increases. However, many of these regions of intersection are not interesting because they are dominated by intersections of a larger number of breakpoint regions. For example, if three breakpoint regions B_i , B_j and B_k have a non-empty intersection, then reporting this intersection is perhaps more desirable

than reporting the (geometrically larger) intersections $B_i \cap B_j$ and $B_i \cap B_k$, particularly as the number of such intersecting regions becomes large. Thus, it is desired to identify regions of intersection of a maximal number of B_i .

We formalize this problem by defining a partial order on intersections of subsets of \mathcal{B} . For $S \subseteq \{1, \dots, n\}$, let $B_{\cap S} = \bigcap_{s \in S} B_s$ denote the intersection of the breakpoint regions indexed by S . Let \mathcal{I}_n be the set of subsets of $\{1, \dots, n\}$ whose corresponding breakpoint regions have non-empty intersection. Formally,

$$\mathcal{I}_n = \{S \subseteq \{1, \dots, n\} \mid B_{\cap S} \neq \emptyset\}. \quad (3)$$

\mathcal{I}_n has the natural partial order of subset inclusion \subseteq , where for two elements I and J of \mathcal{I}_n , $I \prec J$ provided $I \subsetneq J$. We denote this partially ordered set (poset) as $(\mathcal{I}_n, \subseteq)$. We formalize the problem as follows.

MAXIMAL INTERSECTIONS OF BREAKPOINT REGIONS. *Given a set $\mathcal{B} = \{B_1, \dots, B_n\}$ of breakpoint regions, identify all maximal elements of $(\mathcal{I}_n, \subseteq)$.*

Below, we describe how to solve the ‘Maximal Intersections Problem’ by extending the plane sweep algorithm for the ‘All Intersections Problem’.

2.3 Plane sweep algorithm

The plane sweep algorithm was introduced by Shamos and Hoey (1976) for the problem of determining whether n line segments in the plane have any intersections. Clearly this question can be answered in $O(n^2)$ time by checking all pairs of segments for intersection. A plane sweep algorithm performs the same task in $O(n \log n)$ time by first sorting the segments by the x -coordinate of their left endpoint, and then moving the line $x = c$, called the *sweep line* through the plane from left to right. The efficiency of the plane

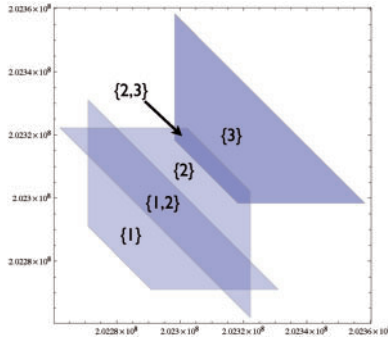


Fig. 2. Breakpoint regions determined by fragments from Kidd *et al.* (2008) whose orientations suggest an inversion variant(s). Breakpoint region 2 has distinct intersections with regions 1 and 3, and thus iterative merging of breakpoint regions will not identify all intersections.

sweep algorithm is derived from two observations. First, not all coordinates c need to be considered. A data structure called the event-point schedule \mathcal{E} records the necessary values of c , and is updated dynamically as the sweep line moves from left to right. Second, for a given position c of the sweep line, the segments intersecting the sweep line can be ordered by the y -coordinate of the intersection. These ordered segments are stored in a data structure called sweep-line status \mathcal{L} . Only adjacent segments in \mathcal{L} need to be examined for intersections. By employing appropriate data structures for \mathcal{E} and \mathcal{L} one obtains an efficient algorithm for segment intersection. Further details of this algorithm can be found in Preparata and Shamos (1985).

The basic framework of the plane sweep algorithm has been extended to numerous related problems in computational geometry such as the counting of the k intersections of n segments in provably optimal $O(n \log n + k)$ time (Chazelle and Edelsbrunner, 1992), and reporting the regions of intersection of polygons in the plane (Nievergelt and Preparata, 1982).

Here, we modify the algorithm of Nievergelt and Preparata (1982) to solve the ‘All Intersections’ and ‘Maximal Intersection’ problems described above. Our extension exploits the particular geometry of the trapezoids and rectangles that define breakpoint regions in order to: (i) efficiently compute their intersection; (ii) label the intersecting regions by the breakpoint regions that are inside; and (iii) iteratively determine the maximal elements of $(\mathcal{I}_n, \subseteq)$.

2.3.1 Overview of the algorithm We provide an overview of the plane sweep algorithm for the case of breakpoint regions defined by inversion variants; i.e. fragments with parallel orientations $(+, +)$ or $(-, -)$. These breakpoint regions are trapezoids with the two parallel sides having slope -1 (Fig. 1A). Thus, we define the sweep line to be a line $y = -x + c$ of slope -1 (Fig. 1A). The sweep line will encounter all inversion breakpoint regions as c increases from c_{\min} to c_{\max} . The algorithm is identical for insertion/deletion variants except the sweep line is chosen to have slope $+1$, $y = x + c$, to match the parallel sides of the trapezoids in this case (Fig. 1B).

As the sweep line advances through the plane, one of three possible events (Fig. 3) can occur: (i) *addition* of a breakpoint region; (ii) *intersection* between two line segments defining the boundaries of breakpoint regions; (iii) *removal* of a breakpoint region (Fig. 3). Since the sweep line is parallel to the two sides of each trapezoid, it is only necessary to consider intersections between the horizontal/vertical sides of the trapezoid. For each breakpoint region B in \mathcal{B} , we define B_{top} and B_{bottom} as the horizontal/vertical sides of the trapezoid. We designate the side with the largest y value as ‘top’.

2.3.2 Data structures As in the plane sweep algorithm for line segment intersection, we maintain two data structures \mathcal{E} and \mathcal{L} . We define the event point schedule \mathcal{E} as the list of positions for the sweep line. The event point schedule \mathcal{E} is initialized with the starts and ends of B_{top} and B_{bottom} ; these correspond to the addition/removal events. \mathcal{E} is updated with new

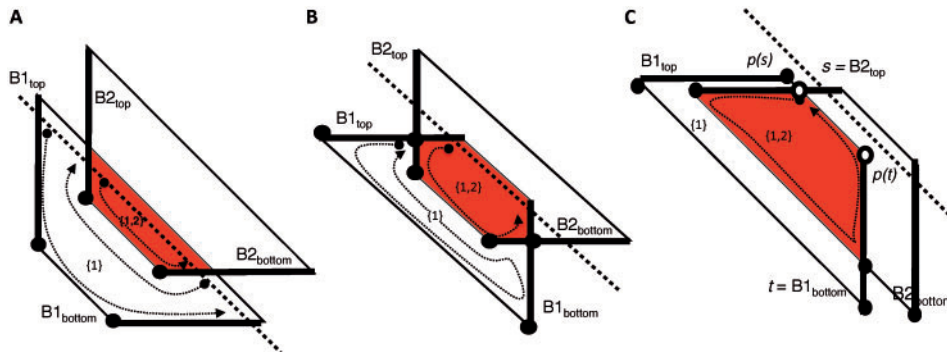


Fig. 3. Examples of the three events of the plane sweep: (A) addition, (B) intersection and (C) removal. In each case black dots label the points recorded in the cyclic lists a and b (indicated as dashed paths) that form \mathcal{R} . In addition, we show in $\{ \}$ the labels assigned to the intersecting breakpoint regions.

Algorithm 1 Plane sweep algorithm

```

Input:  $\mathcal{B}$ , set of breakpoint regions.
Output:  $\mathcal{A}$  all regions of intersection;  $\mathcal{M}$ , regions
        corresponding to maximal elements of  $(\mathcal{I}_n, \subseteq)$ .

//Initialize Data Structures
 $\mathcal{E} :=$  start/end of  $B_{\text{top}}$  and  $B_{\text{bottom}}$  for all  $B \in \mathcal{B}$ 
 $\mathcal{L} := \{-\infty, \infty\}$ ,  $\mathcal{R} := \emptyset$ ,  $\mathcal{H} := \emptyset$ 

//Process All Events
While  $\mathcal{E} \neq \emptyset$ 
     $P := \text{ExtractMin}(\mathcal{E})$ 
     $\text{ProcessEvent}(P, \mathcal{E}, \mathcal{L}, \mathcal{R}, \mathcal{H})$ 
end while

```

intersection points as they are discovered. For a given event point (position of the sweep line), the sweep line status structure \mathcal{L} stores an ordered list of the segments intersecting the sweep line and two terminal segments $y = \pm\infty$. \mathcal{L} is analogous to the same structure in the plane sweep algorithm for line segment intersection. In this case all line segments intersecting the sweep line are either horizontal or vertical edges of breakpoint regions. We first check for intersection between line segments that are adjacent in \mathcal{L} (Fig. 3B). If a non-empty intersection is computed, the intersection point is added to the event schedule \mathcal{E} .

The regions of intersection are recorded using a third data structure \mathcal{R} introduced by Nievergelt and Preparata (1982). \mathcal{R} is attached to the sweep-line status \mathcal{L} and records the vertices of the regions of intersection encountered thus far on the sweep. For each segment s in \mathcal{L} , \mathcal{R} maintains two cyclic lists, $a(s)$ and $b(s)$, that contain the points on the boundaries of the regions above and below s , respectively. Equivalently, if s and t are adjacent line segments in \mathcal{L} , let $[s, t]$ denotes the region to the left of the sweep line and between s and t . Then \mathcal{R} contains the vertices defining the boundary of $[s, t]$. We augment the \mathcal{R} structure of Nievergelt and Preparata (1982) with a label for each region of intersection. This label is the set of breakpoint regions that contain the region of intersection. For example, the region consisting of the non-empty intersection of breakpoint regions B_1 and B_2 is labeled $\{1, 2\}$. Finally, we maintain a interval tree (Preparata and Shamos, 1985) \mathcal{H} from which we derive the maximal elements of the poset $(\mathcal{I}_n, \subseteq)$ encountered thus far on the sweep line.

The algorithm (Algorithm 1) consists of iterating through the event point schedule and updating the regions of intersection found at each step according to whether the event point is an addition, removal, or intersection. This update is briefly described in the next section.

2.3.3 Computing regions of intersection The procedure ProcessEvent in Algorithm 1 updates the data structures, \mathcal{E} , \mathcal{L} , \mathcal{R} and \mathcal{H} according to the type of event. For a removal event, ProcessEvent ends the regions $[s, t]$ for each pair of adjacent segments in \mathcal{L} by joining $b(s)$ and $a(t)$ with the points $p(s)$ and $p(t)$ where s and t intersect the sweep line (Fig. 3C). For an intersection event, ProcessEvent also swaps the order of s and t in \mathcal{L} (Fig. 3B). Further details of these operations are described in (Nievergelt and Preparata, 1982) and in the Supplementary Text (available at <http://www.cs.brown.edu/people/braphael/supplements/structvar>).

Finally, each identified region of intersection is labeled by the constituent breakpoint regions. Region labels are represented as sets of breakpoint region names and are updated using the following procedure. If s and t are consecutive line segments along a sweep line, let $I([s, t])$ denote the label set of the region $[s, t]$. When processing an addition event of breakpoint region i , new regions are introduced with labels $I([s, t]) \cup i$. When a processing a removal event of breakpoint region i , new regions are introduced with label $I([s, t]) \setminus i$. Region labels also change during intersection events. When regions are completed, their labels and list of boundary vertices are inserted

into the interval tree \mathcal{H} . Finally, all regions, or alternatively only maximal regions, are output as they are identified.

2.4 Extensions

We briefly describe two natural extensions of our method. First, we include aCGH data. Second, we compute the probability that a paired-end sequenced fragment matches an existing structural variant.

2.4.1 Incorporating aCGH data As described above, the uncertainty in the breakpoints of a copy number change measured by aCGH is represented as a rectangle (Fig. 3B). The plane sweep algorithm is readily extended to include intersections with the rectangular breakpoint regions.

2.4.2 Incorporating fragment length distribution In most paired-end sequencing approaches, various procedures are used to select fragments of a specified size L , with the resulting fragments having lengths distributed around this selected size. Thus far, we considered each fragment length between L_{\min} and L_{\max} to be equally likely. We can instead derive the empirical distribution $f(L)$ of the values $|y| - |x|$ over all valid pairs (x, y) and use this distribution to better ascertain whether fragments provide evidence for a specific rearrangement. The breakpoint (a, b) defines a length $l_C(a, b) = (\text{sign}(x_C)a - x_C) + (\text{sign}(y_C)b - y_C)$ for fragment C . Given a polygon P defining the breakpoint region of a structural variant, we compute the probability that the invalid pair (x_C, y_C) is consistent with this variant as $\frac{\int_P f(l_C(a, b)) da db}{\int_{G \times G} f(l_C(x, y)) dx dy}$. Note that this length is constant for all points (a, b) on the same line of slope -1 or 1 , according to the orientation of the invalid pair (Bashir et al., 2008).

3 RESULTS

We implemented our geometric approach in a program called GASV. We applied GASV to: (i) analyze recent paired-end sequencing data of nine human individuals; (ii) perform a comparative analysis of genetic structural variants and those identified in paired-end sequencing of several cancer samples; and (iii) integrate data across measurement techniques by comparing variants identified by both aCGH and paired-end sequencing in cancer samples.

3.1 Paired-end sequencing of human structural variants

We used GASV to analyze fosmid paired-end sequencing data from eight individuals from the HapMap populations (Kidd et al., 2008) and another individual from an earlier study (Tuzun et al., 2005). The Kidd et al. (2008) study reported a total of 224 inversion, 724 insertion and 747 deletion variants, which were validated by fingerprint analysis, clone sequencing or FISH. These studies are presently the most comprehensive, high-resolution survey of structural variants in the human genome. The mean insert sizes for the fosmid clones ranged from 36 kb to 41 kb with SD from 1.4 kb to 3.9 kb. In our analysis, we used $L_{\min} = 20$ kb and $L_{\max} = 60$ kb to provide a generous buffer for intersecting breakpoint regions.

3.1.1 Analysis of reported inversion variants We first analyzed the 180 validated inversions reported on the 22 autosomes in Kidd et al. (2008). We obtained the list of the boundaries of each inversion, the names of the clones that support each variant and the mapped coordinates of the end sequences. We used our geometric approach to compute the intersections of the breakpoint regions for each set of supporting clones, and we compared the reported boundaries of the inversions with the intersections we obtained. Surprisingly, 41/180

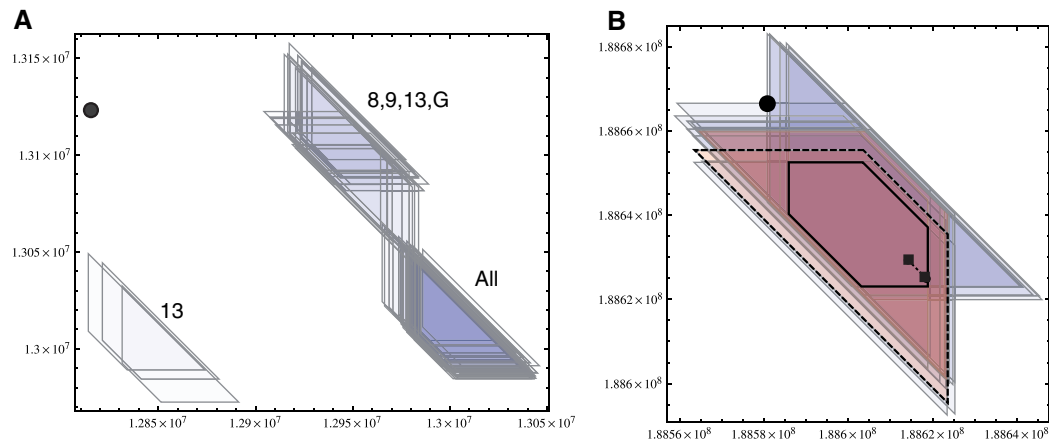


Fig. 4. Geometric analysis of inversion polymorphisms from Kidd *et al.* (2008) reveals disparities between the reported boundary of variants (black dots) and the intersections of breakpoint regions. **(A)** An inversion on chr1 with 79 reported supporting clones from all nine individuals has no point in common to all breakpoint regions. The number x next to each of the three regions indicates a clone from individual labeled ABC x in Kidd *et al.* (2008) is present in the cluster; a ‘G’ indicates the G95 individual from Tuzun *et al.* (2005). The bottom right region contains clones from all nine individuals, while individual ABC13 has clones from all three regions suggesting multiple distinct structural variants or mapping difficulties at this locus. **(B)** An inversion from chr3 with 22 supporting clones from all eight HapMap individuals. We examined one fully sequenced clone (dashed trapezoid) from individual ABC7 and found two possible inversion breakpoints (black squares). Both of these lie in the intersection of all breakpoint regions but are ~ 37 kb from the reported boundary.

of the validated inversions had an empty intersection of breakpoint regions. That is, there were no candidate inversion breakpoints common to all of the reported supporting clones suggesting that the mapped clones are inconsistent with only a single inversion at the locus. Figure 4A shows an example of one such set, where multiple, distinct non-overlapping intersections are visible. One hypothesis is that the three distinct regions of intersection might represent slightly different breakpoints in different individuals. However, one individual contains clones from all three regions, suggesting that this genomic locus harbors a more complex rearrangement.

In the remaining 139 cases the reported boundaries were not in the region of intersection. Figure 4B shows one example where the reported coordinates for an inversion are clearly outside the region of intersection. In this case, Kidd *et al.* (2008) sequenced one of the clones in this cluster. We aligned this sequence to the reference genome and obtained two possible inversion breakpoints, both of which lie in the region of intersection computed by GASV. These two breakpoints could not be further resolved due to repetitive sequence near the inversion breakpoints. Analysis of additional sequenced clones from Kidd *et al.* (2008) showed a number of additional inversion breakpoints that occur within segmental duplications. Thus even with complete sequence data available, resolving the breakpoint with greater precision is challenging.

The method used to derive the reported boundaries of the variants in Kidd *et al.* (2008) is mysterious. It is possible that the reported boundaries were intended to represent a ‘consensus’ of a single structural variant locus. For complicated loci with multiple, overlapping rearrangements (Fig. 4A), consensus coordinates might provide a reasonable summary of the data. However, we find that in many cases, the data allow us to refine the breakpoint region, and that significant information is lost when only a single pair of coordinates is reported for an inversion.

3.1.2 Analysis of intersecting breakpoint regions Using the complete set of mapped locations provided by Kidd *et al.* (2008),

we computed the intersections of breakpoint regions for all nine individuals using GASV. In total, 30 853 clones on the 22 autosomes were consistent with an inversion. There were 1361 groups of intersecting breakpoint regions. Of these, 1200 had non-empty intersections, indicating that these clusters have a putative breakpoint in common for all of the clones. The remaining 161 groups had no breakpoints common to all the clones. Thus, over 10% the intersecting breakpoint regions suggest either complicated loci that are not easily explained as single inversion variants, or mapping/alignment artifacts.

For the 1200 clusters with non-empty intersection, we computed the area of the intersection and defined the *localization* of a breakpoint as the square root of this area. Thus, if the region of intersection was a square, the localization would give the genomic range allowed for each breakpoint. There are 19 clusters with breakpoint localization < 2500 bp. In the best example, eight clones from four individuals localize the breakpoints to within 286 bp on each end (Supplementary Text). This is a remarkably small region of uncertainty, considering that a single fosmid clone localizes a breakpoint to ≈ 40 kb.

We also found that breakpoint localization is not directly correlated with the number of clones in the breakpoint region. In 21/1200 cases the breakpoint region was supported by more than 50 clones. In only two of these cases was the breakpoint localization < 5000 bp. A possible reason for this discrepancy is the presence of repeats/duplications near the inversion breakpoints. These would lead to a relatively small genomic region where end sequences can be mapped uniquely.

3.1.3 Overlap of inversion and deletion variants We used GASV to compare the locations of inversions and deletions in the data of Kidd *et al.* (2008). We identified 5054 instances of intersection between inversion and deletion breakpoint regions. Figure 5 shows a sample cluster containing 33 clones indicating an inversion and

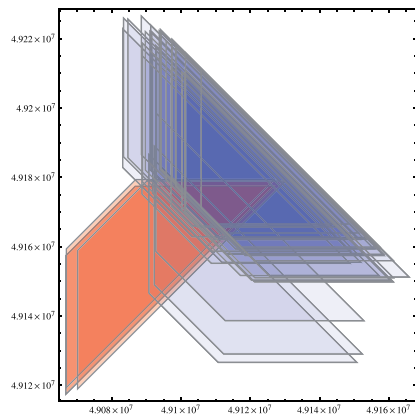


Fig. 5. Intersection of 33 inversion breakpoint regions (blue) and 4 deletion breakpoint regions (red), indicates common genomic location of two structural variants.

4 clones indicating a deletion at the same locus. There are several examples where inversion heterozygotes lead to deletions in progeny (Stankiewicz and Lupski, 2002), a possible explanation for these overlapping variants. Alternatively, this overlap might suggest that these regions are unstable and subject to repeated rearrangement (Stankiewicz and Lupski, 2002).

3.2 Cross-study comparison of structural variants

Our geometric approach allows for the comparison of structural variants identified in different individuals with different measurement techniques. We tested this feature by comparing the genetic structural variants identified by Kidd *et al.* (2008) with variants identified in ESP studies of cancer genomes (Raphael *et al.*, 2008; Volik *et al.*, 2006). The later studies aimed to identify somatic, and possibly cancer-related, rearrangements in three breast cancer cell lines and five primary tumors from various tissues. The cancer ESP studies used large insert clones (BACs) with average sizes of 150 kb. We first identified clusters of invalid pairs in each cancer dataset that were suggestive of either inversions or deletions, and then computed the intersection of these clusters with the inversion and deletion clusters computed from the nine normal individuals. Approximately 5–53% of invalid clusters from the cancer clusters are consistent with inversion or deletion variants identified in normal individuals (Table 1). The larger percentages are found in the primary tumor samples; this is consistent with the lower sequence coverage in the primary tumor samples, and the fact that tumor samples frequently contain significant admixture of normal cells resulting from difficulty of separating normal from tumor cells.

We then clustered all the cancer data together with the nine normal individuals, and identified overlapping breakpoint regions containing at least two invalid pairs from different cancer samples. Of the 22 such clusters, 10 are consistent with inversion variants identified in at least one normal individual (9/10 cases were observed in at least four normal individuals), demonstrating that a large fraction of structural variants found in more than one cancer dataset are inherited genetic variants and not somatic rearrangements.

Table 1. A comparison of the inversion and deletion variants identified in nine normal individuals (Kidd *et al.*, 2008; Tuzun *et al.*, 2005) and several cancer genomes (Raphael *et al.*, 2008; Volik *et al.*, 2006)

Cancer sample	No. of concordant inversions (%)	No. of concordant deletions (%)
MCF7	8 (5)	40 (28)
BT474	12 (19)	8 (11)
SKBR3	8 (13)	7 (11)
Breast	11 (19)	21 (27)
Breast	12 (32)	19 (38)
Prostate	3 (9)	12 (27)
Ovary	8 (53)	12 (29)
Brain	2 (11)	10 (26)

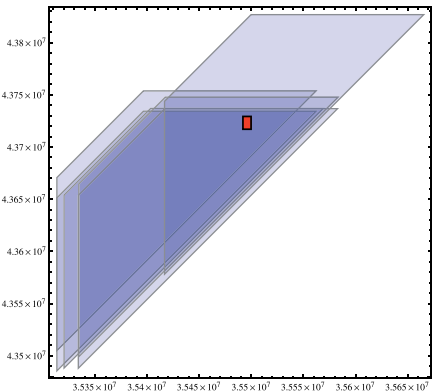


Fig. 6. Intersection between six breakpoint regions from ESP data (blue trapezoids) and two breakpoint regions determined by aCGH (red rectangle) on chr17 in the BT474 breast cancer cell line. In this case, the spacing between aCGH probes provides a more precise localization of the breakpoint region that the paired-end sequencing data.

3.3 Comparing variants identified by paired-end sequencing and aCGH

We used GASV to compare the breakpoints identified by ESP and aCGH for three cancer cell lines, MCF7, BT474 and SKBR3, using data from Volik *et al.* (2006), Raphael *et al.* (2008), and Aerni *et al.* (2009). We formed rectangles corresponding to pairs of copy number changes identified by segmentation (Fig. 1B) of aCGH data using CBS (Olshen *et al.*, 2004). We found that 35/152, 20/380 and 35/149 of the clusters defined from paired-end sequenced data intersected aCGH breakpoint regions in BT474, MCF7 and SKBR3, respectively. Figure 6 shows an example of a cluster containing 19 breakpoint regions identified by ESP in the BT474 cell line, intersecting a breakpoint region determined by aCGH.

4 DISCUSSION

We introduced GASV, a geometric approach for classification and comparison of structural variants. To our knowledge, this is the first comprehensive method for structural variant analysis across multiple samples that supports both paired-end sequencing data with arbitrary fragment sizes and aCGH with varying array resolutions.

We illustrated the generality of our approach through several applications, including the clustering of variants from a paired-end sequencing study of nine individuals, the comparison of variants in normal and cancer genomes derived through different sequencing approaches, and the comparison of variants identified by aCGH and paired-end sequencing of the same cancer samples. In many cases we are able to localize the breakpoints of single variants, but in other cases the end-sequence pairs suggest more complicated variants. The precise localization of the boundaries of structural variants provided by the GASV is helpful for distinguishing simple variants shared across multiple individuals from more complex variants resulting from repeated rearrangements at the same locus. These results also demonstrate the importance of identifying and reporting the uncertainty in structural variant boundaries. The current convention of publishing approximate coordinates that were derived from study-specific heuristics can lead to unnecessary errors and misannotations of complicated variants.

We expect that GASV will be useful for analyzing data from the 1000 Genomes Project and for cancer genome sequencing efforts that are part of The Cancer Genome Atlas. In the latter application, GASV will help distinguish genetic from somatic rearrangements.

There are several directions for future work. First, it would be useful to perform a more comprehensive comparison of the variants that are identified by different measurement techniques. aCGH has limited power to detect variants whose breakpoints lie in repeat-rich regions of the genome due to the inability to identify probes in these regions. Paired-end sequencing approaches can be similarly limited, particularly if small fragment sizes are used, since the end sequences will not align uniquely to repeat-rich regions of the genome. Current studies of structural variants with next-generation sequencing technologies have used small fragment sizes from 200 bp (Campbell *et al.*, 2008) to 3 kb (Korbel *et al.*, 2007). An unresolved question is the optimal fragment size to use for studies of human structural variation. We have shown that clustering of breakpoint regions from relatively large clones (40 kb) with GASV can yield very precise localization of variant breakpoints (a few hundred base pairs). Kidd *et al.* (2008) reported that most of the clones that they sequenced had highly repetitive sequence at the breakpoints, complicating the precise breakpoint identification and assembly of the clone sequence. Thus, even in cases where complete sequence is available GASV can be used to record uncertainty in breakpoint location.

An additional area of future work is to incorporate breakpoint uncertainty into databases of known structural variants. Our geometric approach could then be used to query this database and thus provide a more robust procedure for comparing newly discovered and existing variants. In addition, knowledge of existing structural variants can be used to guide mapping of end sequences that do not map uniquely to the reference genome. This is a common problem in human genome resequencing, where up to 60% percent of ES fragments are not used because of their ambiguous mappings (Korbel *et al.*, 2007). Lee *et al.* (2008) recently described a probabilistic model for resolving ambiguities that arise when mapping ES pairs in a single sample. Developing a model for multi-sample comparison that incorporates variant ambiguity across samples is a promising future endeavor.

Finally, we focused exclusively on structural variation in the human genome, but such variation is also found in the mouse genome (Egan *et al.*, 2007) and other model organisms (Dopman and

Hartl, 2007). Thus, there will continue to be an increasing demand for better analysis tools for structural variation.

ACKNOWLEDGEMENTS

We thank Franco Preparata and Crystal Kahn for helpful technical discussions, and Anna Ritz for assistance in preparing the manuscript.

Funding: Career Award at the Scientific Interface from the Burroughs Wellcome Fund (to B.J.R.); the Department of Defense Breast Cancer Research Program (to B.J.R.); ADVANCE Program at Brown University, which is funded by the National Science Foundation under grant number 0548311 (to B.J.R.).

Conflict of Interest: none declared.

REFERENCES

- Aerni, S. *et al.* (2009) Combined analysis of copy number changes and structural rearrangements in cancer genomes.
- Bashir, A. *et al.* (2008) Evaluation of paired-end sequencing strategies for detection of genome rearrangements in cancer. *PLoS Comput. Biol.*, **4**, e1000051.
- Campbell, P. *et al.* (2008) Identification of somatically acquired rearrangements in cancer using genome-wide massively parallel paired-end sequencing. *Nat. Genet.*, **40**, 722–729.
- Chazelle, B. and Edelsbrunner, H. (1992) An optimal algorithm for intersecting line segments in the plane. *J. ACM*, **39**, 1–54.
- Conrad, D. *et al.* (2006) A high-resolution survey of deletion polymorphism in the human genome. *Nat. Genet.*, **38**, 75–81.
- Cooper, G. *et al.* (2008) Systematic assessment of copy number variant detection via genome-wide SNP genotyping. *Nat. Genet.*, **40**, 1199–1203.
- Dopman, E. and Hartl, D. (2007) A portrait of copy-number polymorphism in *Drosophila melanogaster*. *Proc. Natl Acad. Sci. USA*, **104**, 19920–19925.
- Egan, C. *et al.* (2007) Recurrent DNA copy number variation in the laboratory mouse. *Nat. Genet.*, **39**, 1384–1389.
- Emerson, J. *et al.* (2008) Natural selection shapes genome-wide patterns of copy-number polymorphism in *Drosophila melanogaster*. *Science*, **320**, 1629–1631.
- Fridlyand, J. *et al.* (2004) Hidden markov models approach to the analysis of array CGH data. *J. Multivar. Anal.*, **90**, 132–153.
- Iafraite, A. *et al.* (2004) Detection of large-scale variation in the human genome. *Nat. Genet.*, **36**, 949–951.
- Kidd, J. *et al.* (2008) Mapping and sequencing of structural variation from eight human genomes. *Nature*, **453**, 56–64.
- Korbel, J.O. *et al.* (2007) Paired-end mapping reveals extensive structural variation in the human genome. *Science*, **318**, 420–426.
- Lee, S. *et al.* (2008) A robust framework for detecting structural variations in a genome. *Bioinformatics*, **24**, 59–67.
- Levy, S. *et al.* (2007) The diploid genome sequence of an individual human. *PLoS Biol.*, **5**, e254.
- Marshall, C. *et al.* (2008) Structural variation of chromosomes in autism spectrum disorder. *Am. J. Hum. Genet.*, **82**, 477–488.
- McCarroll, S. *et al.* (2008) Integrated detection and population-genetic analysis of SNPs and copy number variation. *Nat. Genet.*, **40**, 1166–1174.
- Myers, C. *et al.* (2004) Accurate detection of aneuploidies in array CGH and gene expression microarray data. *Bioinformatics*, **20**, 3533–3543.
- Nievergelt, J. and Preparata, F.P. (1982) Plane-sweep algorithms for intersecting geometric figures. *Commun. ACM*, **25**, 739–747.
- Olshen, A.B. *et al.* (2004) Circular binary segmentation for the analysis of array-based DNA copy number data. *Biostatistics*, **5**, 557–572.
- Perry, G. *et al.* (2008) The fine-scale and complex architecture of human copy-number variation. *Am. J. Hum. Genet.*, **82**, 685–695.
- Pinkel, D. and Albertson, D.G. (2005) Array comparative genomic hybridization and its applications in cancer. *Nat. Genet.*, **37** (Suppl.), S11–S17.
- Pinkel, D. *et al.* (1998) High resolution analysis of DNA copy number variation using comparative genomic hybridization to microarrays. *Nat. Genet.*, **20**, 207–211.
- Preparata, F.P. and Shamos, M.I. (1985) *Computational Geometry—An Introduction*. Springer.

- Raphael,B. *et al.* (2008) A sequence-based survey of the complex structural organization of tumor genomes. *Genome Biol.*, **9**, R59.
- Raphael,B.J. *et al.* (2003) Reconstructing tumor genome architectures. *Bioinformatics*, **19** (Suppl. 2), II162–II171.
- Redon,R. *et al.* (2006) Global variation in copy number in the human genome. *Nature*, **444**, 444–454.
- Scherer,S. *et al.* (2007) Challenges and standards in integrating surveys of structural variation. *Nat. Genet.*, **39**, 7–15.
- Sebat,J. *et al.* (2004) Large-scale copy number polymorphism in the human genome. *Science*, **305**, 525–528.
- Shamos,M.I. and Hoey,D. (1976) Geometric intersection problems. In *FOCS*, pp. 208–215.
- Sharp,A.J. *et al.* (2006) Structural variation of the human genome. *Annu. Rev. Genomics Hum. Genet.*, **7**, 407–442.
- Stankiewicz,P. and Lupski,J. (2002) Genome architecture, rearrangements and genomic disorders. *Trends Genet.*, **18**, 74–82.
- Tuzun,E. *et al.* (2005) Fine-scale structural variation of the human genome. *Nat. Genet.*, **37**, 727–732.
- Volik,S. *et al.* (2003) End-sequence profiling: sequence-based analysis of aberrant genomes. *Proc. Natl Acad. Sci. USA*, **100**, 7696–701.
- Volik,S. *et al.* (2006) Decoding the fine-scale structure of a breast cancer genome and transcriptome. *Genome Res.*, **16**, 394–404.
- Wheeler,D. *et al.* (2008) The complete genome of an individual by massively parallel DNA sequencing. *Nature*, **452**, 872–876.